Elite EL

Remote Update Tools

Developer Guide for Initialization Tool v1.0.0.1

SENSESHIELD

COPYRIGHTS AND TRADEMARKS

The Elite EL® with its technical documentation is copyrighted (C) 2017 to present by Senseshield Technology Co., Ltd (Senseshield). All rights reserved.

All products referenced throughout this document are trademarks of their respective owners.

All attempts have been made to make the information in this document complete and accurate. Senseshield is not responsible for any direct or indirect damages or loss of business resulting from inaccuracies or omissions. The specifications contained in this document are subject to change without notice.

CONTACT

Company Name: Beijing Senseshield Technology Co., Ltd

Address: Suite 510,Block C,Internet Innovation Center,Building 5,
No.10,Xibeiwang East Road,Haidian District,
Beijing China

Tel.: +86-10-56730936
Fax: +86-10-56730936-8007

Sales: sense@sense.com.cn
Website: www.sense.com.cn

LICENSE AGREEMENT

PLEASE READ THIS AGREEMENT CAREFULLY BEFORE USING THE CONTENTS THEREOF AND/OR BEFORE DOWNLOADING OR INSTALLING THE SOFTWARE PROGRAM. ALL ORDERS FOR AND USE OF THE Elite AND/OR EL FAMILY PRODUCTS (including but not limited to the Kit, libraries, utilities, diskettes, disc, Senseshield® and/or Senseshield® keys, the software component of Senseshield and/or EL and the EL License Guide) (hereinafter “Product”) SUPPLIED BY Senseshield Technology Co., Ltd (hereinafter "Senseshield") ARE AND SHALL BE, SUBJECT TO THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT.

This document is a legally binding agreement between you (either an individual or an entity) and Senseshield®. If you are not willing to be bound by the terms of this agreement, you should promptly (and at least within 3 days from the date you received this package) return the unused developer's kit and the programmer's guide to Senseshield. Use of the software indicates your acceptance of these terms.

- **GRANT OF LICENSE**

  The software of the Product is being licensed to you, which means you have the right to use the software only in accordance with this License Agreement. You may (a) copy the software for internal use, (b) modify the software for the purpose of integrating with your application and (c) merge the software with other programs.

- **NON-PERMITTED USES**

  Except explicitly permitted in this License Agreement, you may not (a) copy, modify, reverse engineering, decompose, assemble the Product in whole or in part, or (b) sell, lease, license, transfer, distribute all or part of the Product or rights granted in this License Agreement.

- **LIMITED WARRANTY**

  After the date of purchase, Senseshield provides 24-month warranty that the Senseshield EL key has no material and manufacturing defects substantially. All the responsibilities of Senseshield Technology Co., Ltd and all the compensation you can get under warranty are: you can require replace/repair the Product or accept other remedial measures.

- **LIMITATION OF LIABILITY**

  Under any circumstances, Senseshield will NOT be liable for any damages arising out of usage or inability of the Product, including but not limited to: loss of data, loss of profits, and other special, incidental, joint, secondary or indirect loss.

  Except for the limited warranty offered to the original buyer, Senseshield is not responsible for providing any insurance to anyone on the product, performance and service including merchantability and fitness for a particular purpose.

  The entire product, including Senseshield EL, the software, the document, other material shipped as accessories, and backups made by you are copyrighted by Senseshield Security GmbH.

- **TERMINATION**

  Your failure to comply with the terms of this License Agreement shall terminate your license and this License Agreement.

CONTENTS

COPYRIGHTS AND TRADEMARKS ..... **I**

CONTACT ..... **II**

LICENSE AGREEMENT ..... **III**

CONTENTS ..... **IV**

OVERVIEW **1**

About the Guide 1
What are EL Remote Update Tools? 1

INITIALIZATION TOOL **2**

What is Initialization Tool? 2
User Environment 2
Directories 2
Interface 3

HOW TO USE **7**

Preserve the Key 7
Create User Dongle 7
Create Issuer Dongle 7
Release File 7
Error Information of Operating Devices 8

INITIALIZATION LIB **9**

What is Initialization Lib? 9
Explanation on Header File 10
API of Initialization Lib 13
e4ru_CreateIssuerDevice *13*
e4ru_CreateClientDevice *14*
e4ru_ReleaseFile *16*

OVERVIEW 1

About the Guide

Mode	Model	Version	Releasing Date
Elite EL Remote Update Tools Initialization Tool	STD, Genii, RTC, RTCC, NET	v1. 0.0.1	2011.01.29

- **Conventions Used**

The following conventions are used throughout this document:

Italic	Words in italic represent file names and directory names.
Bold	Words in boldface represent keystrokes, menu items, and window names and fields.
	The caution icon flags some content you should be careful of.

- **Document Improvement**

Document Writing Team dedicates to insure the accuracy and completeness of context. Your feedback will assist them to make continuous improvement on EL document. Please do not hesitate to email us, sense@sense.com.cn.

What are EL Remote Update Tools?

The EL Remote SDK contains three tools: *Initialization Tool*, *Issuer Tool* and *User Update Tool.*

- **Initialization Tool**

InitTool.exe runs at the developer end, mainly support functions to **initialize Issuer Device** (Issuer Key) and **initialize User Device** (User Key), as well as **release information of User Device.**

- **Issuer Tool**

IssuerTool .exe is mainly used by software vendors. This tool is used to **create data module** and **generate update packages.**

- **User Update Tool**

ClientTool.exe runs at the user end. With this tool, users can **review internal information of device, review information of Update Package** and **update User Device** (User Key)**.**

INITIALIZATION TOOL 2

What is Initialization Tool?

InitTool.exe runs at the developer end, mainly support functions to initialize the Issuer Dongle (Issuer Device), initialize the User Dongle (Client Device) and release the information of User Dongle

- **Initialize Issuer Dongle:**

Issuer makes Issuer Dongle by initialization. Developer can make update package by using this Issuer Dongle

- **Initialize User Dongle:**

Issuer makes User Dongle by initialization. The User Dongle can be remotely updated by using the update package made by the Issuer Dongle.

- **Release File:**

Using this tool can release and store the hardware program making Issuer Dongle and User Dongle with the information file of latter to specified directory. The default file names are FB01.bin, FB11.bin and FB12.dat.

User Environment

OS Supported: Windows NT4.0, Windows 98 2nd, Windows ME, Windows 2000, Windows XP and Windows 2003.

Directories

- **Dll: E4RUInit.dll, dongle.dll, GetRandom.dll.**
- **Language: language.dll.**
- **Initiation Tool: *InitTool.exe*.**
- **File Type Conversion Tool (*.bin to*.hex): *BinToHex_Tool.exe*.**
- **Header file corresponding to E4RUInit.dll: E4RUInit.h, dongle.h.**
- **Lib file of output interface of *E4RUInit.dll: E4RUInit.lib*.**

Interface

Before running, the program will analyze the config file (*initTool.ini*) of key TDesKey which is used to encrypt and decrypt. If this file is not existed, the program will prompt the following window **save shared secret key**.



Figure 1

The field Shared secret key shows the key TDesKey, 32 characters, represented in hex.

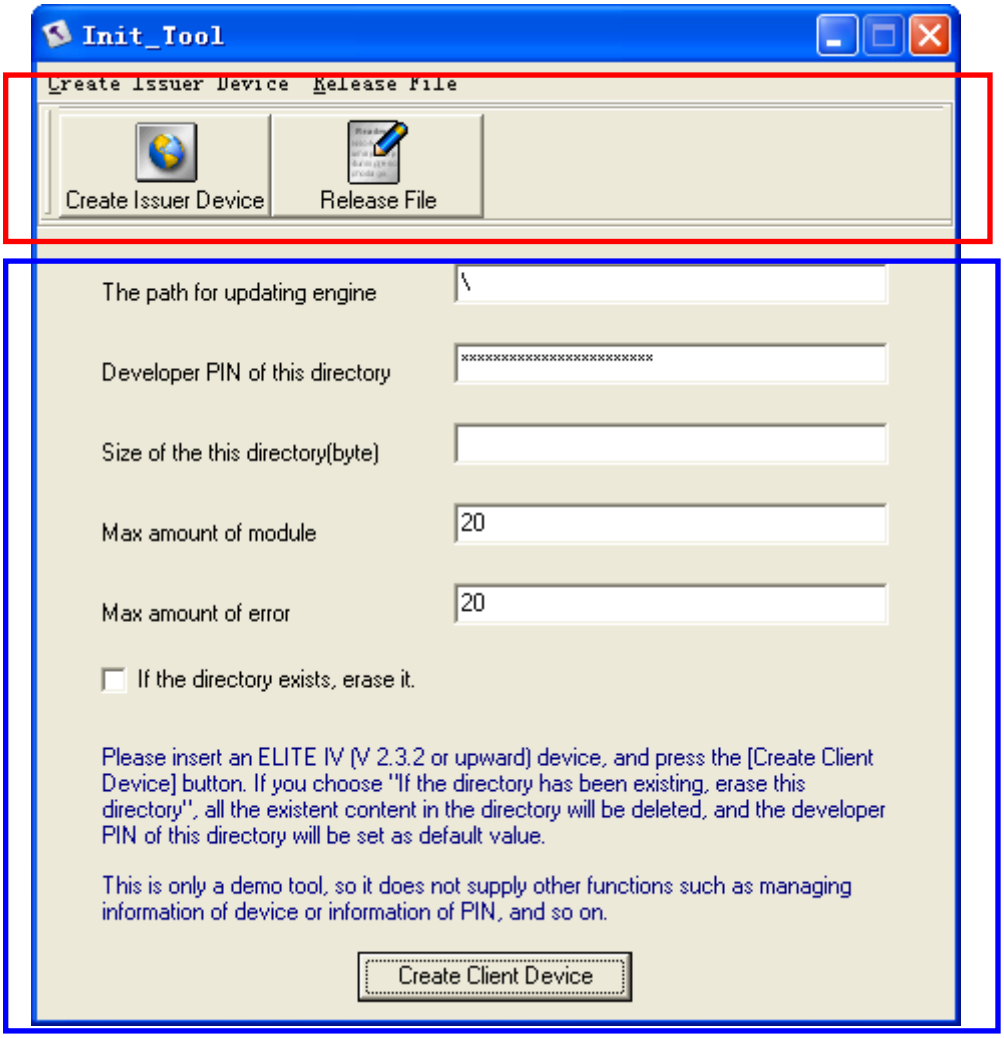


Figure 2

Square in red, is the shortcut functioning area, click to pop out the windows of **Create Issuer Device** and **Release File respectively**.

Square in blue, is the functioning area to create the User Dongle.

- **The path of updating engine:**
  It is used to store the path of updating engine file for User Dongle, only supporting operations on root directory and 1st level directory. The format is '\', '\xxxx'. When storage directory is 1st level, namely '\xxxx', the Directory ID must be 4 characters in hex. The default path is '\', namely the root directory. If the User Dongle to be made is the network version, this parameter must be '\'.

- **Developer PIN of this directory:**
  The Developer ID of the directory for storing the updating engine file, 24-byte long, "122345678123456781234567 8" in default.

  When the specified directory of storing updating engine file does not exist:

1. If the directory is root directory, its input value is "122345678123456781234567 8". After the successful creation, the Developer PIN of root directory should be default.
2. If the directory is sub directory, the input value should be the Developer PIN of upward directory. After the successfully creation, its Developer PIN should be default.

   When the specified directory of storing updating engine file exists:

3. If the checkbox if the directory exists, erase it is ticked, after the successful creation, the Developer PIN of the specified directory is the default value.
4. If it is ticked, after the successful creation, the Developer PIN of the specified directory will not change.

- **Size of this directory**
  When the updating engine file of User Dongle is in the root directory, this value could be Null; in the subdirectory, the value could not be Null, and recommended 5120 bytes in min.

- **Max amount of module**
  The max amount of module is allowed by the remote updating function, ranging 1~ 255 and 20 in default.

- **Max amount of error**
  The max amount of error is allowed by the remote updating function, ranging 1~ 255 and 20 in default.

- **if the directory exists, erase it is ticked,**
  In case of downloading updating engine, ticking this checkbox, if the specified directory does exist, will empty first and then start downloading; unticking will overwrite the file. If the User Dongle to be created is network version, this operation is non-effective.

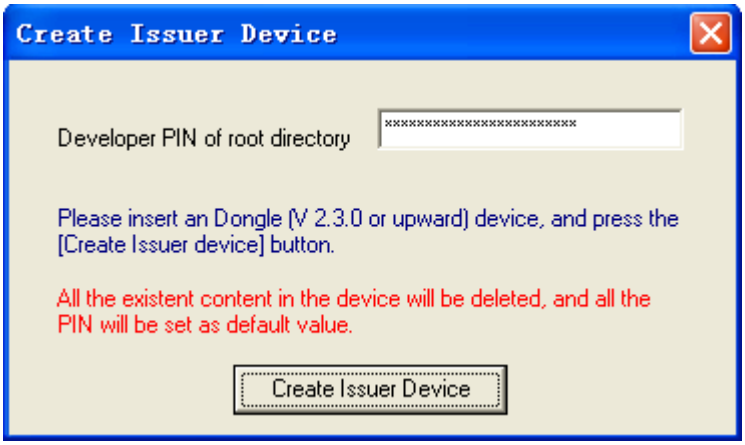


Figure 3

The input field Developer PIN of root directory is the one of the Issuer Dongle, 24-byte long.

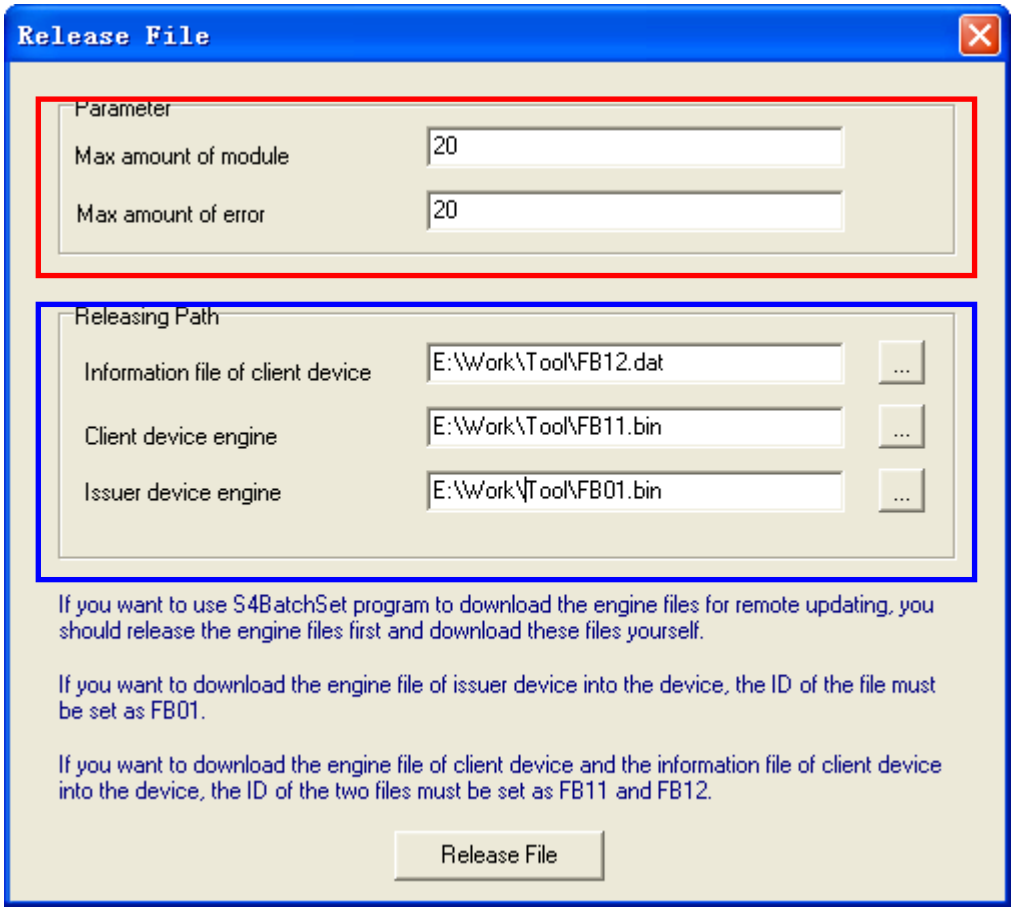


Figure 4

The square in red is the input area of operating parameters to release the file.

- **Max amount of module**
  The max amount of module is allowed by the remote updating function, ranging 1~ 255 and 20 in default.

- **Max amount of error**
  The max amount of error is allowed by the remote updating function, ranging 1~ 255 and 20 in default.

The square in blue is the input field of the path storing the releasing file, Information File of client device, Client device engine and Issuer device engine. You could set up them according

to your request; the default value is the current path.

How to Use

3

Preserve the Key

TDeskey is to use in TDES encryption and decryption for remote updating engine. Due to security, TDeskey is to store in the file InitTool.ini under the directory of tool in form of key value. (non-encrypted)

Create User Dongle

After the verification of InitTool.ini, the main interface works out afterwards. All parameters must be input correctly according to the requirement; otherwise the error will be prompted. And the program will retrieve TDesKey from *InitTool.ini*.

Click Create Client Device



If the subdirectory is created in the process of creating User Dongle, it is recommended to change its Developer PIN instead default value.

Create Issuer Dongle

Click **Create Issuer Device** or icon to get a new window asking for valid Developer PIN of directory.

Click **Create Issuer Dongle**. TDesKey will be retrieved from InitTool.ini. It will return successful message if succeeded, or error code.



In the process of creating Issuer Dongle, the root directory will be deleted and rebuilt, it is recommended to change its Developer PIN instead default value.

Release File

Click **Release File** button or icon to get into window, input the parameters correctly and specify the releasing path.

Click **Release File**, TDesKey will be retrieved from *InitTool.ini*. It will return successful message if succeeded, or error code.



If you would like to use the DevTest to download the released file into the device, please tick the checkbox readable-writable in downloading the file FB11, otherwise, it will be failed.

Error Information of Operating Devices

The error information is mainly designed for dynamic API:

Item	Solution
Failed to find device	Please check the device is correctly connected or not
Invalid Hardware version	Please check the hardware version is valid or not
Failed to open device	Check the device is correctly connected or not, or some program is using the device in exclusive mode.
Failed to verify the Developer PIN	Please review out the Developer PIN of the directory storing the updating engine
Insufficient storage of specified directory	If the directory is existed, check out the remaining space will work out or not, it is recommended to empty or create a new one; if the directory is new, check its size is available or not, it is recommended to rebuild and increase the memory space.
Ineffective directory name	Please review its naming method is valid or not.
updating file out of boundary of	A file sharing the same name with the updating engine is existed in the specified directory; its size is insufficient for expected file to be input. Emptying or creating new one is recommended.
File type mismatch	A file sharing the same name with the updating engine is existed in the specified directory, their types do not match. Emptying or creating new one is recommended.
Hardware type mismatch	The connected device is not local type; please check out its hardware type.

INITIALIZATION LIB 4

What is Initialization Lib?

The initialization lib provides functions to initialize the Issuer Dongle, the User Dongle and release the file to enable developers to integrate the updating function into their software. It consists of *E4RUInit.dll*, *E4RUInit.lib*, *E4RUInit.h* and *dongle.h*.

You could connect *E4UInit.dll* invisibly by *E4RUInit.lib*, or visibly load *E4RUInit.dll*.

E4RUinit.dll supports the customization development; this guide is to help you use the function more effectively.

In the customization project, it is required to invoke *dongle.dll* in process of invoking initialization lib. You have to use following methods:

5. Using VC compiler, *dongle.lib* and *E4RUInit.lib* can be just added into the project.
6. Using other compiler (C++ Builder or Delphi), dongle.dll and E4RUInit.dll must be invoked, otherwise calling exception will occur.

Explanation on Header File

E4RUInit.h is as following:

```
#ifndef __E4RU_INIT_H__
#define __E4RU_INIT_H__

#ifdef __cplusplus
extern “C” {
#endif

#if defined WIN32 || defined _WIN32 || defined _WIN64
#include <windows.h>
#endif

#include “dongle.h”

//* Return value definition */

#define E4RU_SUCCESS                            0x00000000

#define E4RU_INVALID_PARAMETER                  0xE0002002
#define E4RU_INSUFFICIENT_BUFFER                0xE0002003
#define E4RU_DEVICE_NOT_FOUND                   0xE0002004

#define E4RU_INVALID_DEVICE_VERSION             0xE0002101
#define E4RU_OPEN_DEVICE_ERROR                  0xE0002102
#define E4RU_INSUFFICIENT_MEMORY                0xE0002103
#define E4RU_CREATE_DIR_ERROR                   0xE0002104
#define E4RU_ERASE_DIR_ERROR                    0xE0002105
#define E4RU_VERIFYPIN_ERROR                    0xE0002106
#define E4RU_WRITEFILE_ERROR                    0xE0002107
#define E4RU_INSUFFICIENT_DIR                   0xE0002108
#define E4RU_INVALID_DIRNAME                    0xE0002109
#define E4RU_UPDATE_FILE_OVERSIZE               0xE000210A
#define E4RU_GET_FILE_INFO_ERROR                0xE000210B
#define E4RU_MISMATCH_FILE_TYPE                 0xE000210C
#define E4RU_GET_DEVICE_TYPE_ERROR              0xE000210D
#define E4RU_MISMATCH_DEVICE_TYPE               0xE000210E
#define E4RU_GET_AVAILABLE_SPACE_ERROR          0xE000210F

//* flag value definition */

#define E4RU_ERASE_DIR                          0x00000000
#define E4RU_UPDATE_FILE                        0x00000001

//* Function declaration*/

/*
  Create the issuer device.

  @parameter    pS4Ctx          [in]  The context pointer of the device.
  @parameter    pDevPin         [in]  Developer pin value, 24 byte.
  @parameter    pTDESA          [in]  Shared TDES Key.

  @return value
  If the function succeeds, the return value is E4RU_SUCCESS, otherwise return
other defined return value.
*/
DWORD WINAPI e4ru_CreateIssuerDevice(
        IN      PDONGLE_CONTEXT         pS4Ctx,
        IN      BYTE                    *pDevPin,
        IN      BYTE                    *pTDESA
);

/*
  Create the client device.
```

```
  @parameter    pS4Ctx            [in]  The context pointer of the device.
  @parameter    lpDirectory       [in]  ID of the specified directory.
  @parameter    Size              [in]  Size of the specified directory.
  @parameter    dwEraseFlag       [in]  Flag of the erase directory.
  @parameter    pDevPin           [in]  Developer pin value, 24 byte.
  @parameter    pTDESA            [in]  Shared TDES Key.
  @parameter    ModuleNum         [in]  Max counts of the module.
  @parameter    ErrorMax          [in]  Max counts of the error.

  @return value
  If the function succeeds, the return value is E4RU_SUCCESS, otherwise return
other defined return value.
*/
DWORD WINAPI e4ru_CreateClientDevice(
        IN        PDONGLE_CONTEXT        pS4Ctx,
        IN        LPCSTR                 lpDirectory,
        IN        DWORD                  Size,
        IN        DWORD                  dwEraseFlag,
        IN        BYTE                   *pDevPin,
        IN        BYTE                   *pTDESA,
        IN        BYTE                   ModuleNum,
        IN        BYTE                   ErrorMax
);

/*
  Release the files.

  @parameter    pTDESA            [in]  Shared TDES Key.
  @parameter    ModuleNum         [in]  Max counts of the module.
  @parameter    ErrorMax          [in]  Max counts of the error.
  @parameter    pdwIssuerLen [in][out]  Size of the issuer device's binary file.
  @parameter    pdwClientLen [in][out]  Size of the client device's binary file.
  @parameter    pdwInfoLen   [in][out]  Size of the device's information file.
  @parameter    pIssuerContent   [out]  Content of the issuer device's binary file.
  @parameter    pClientContent   [out]  Content of the client device's binary file.
  @parameter    pInfoContent     [out]  Content of the device's information file.

  @return value
  If the function succeeds, the return value is E4RU_SUCCESS, otherwise return
other defined return value.
*/
DWORD WINAPI e4ru_ReleaseFile(
        IN        BYTE                   *pTDESA,
        IN        BYTE                   ModuleNum,
        IN        BYTE                   ErrorMax,
        IN OUT    DWORD                  *pdwIssuerLen,
        IN OUT    DWORD                  *pdwClientLen,
        IN OUT    DWORD                  *pdwInfoLen,
        OUT       BYTE                   *pIssuerContent,
        OUT       BYTE                   *pClientContent,
        OUT       BYTE                   *pInfoContent
);

#ifdef __cplusplus
}
#endif // extern “C”

#endif // __E4RU_INIT_H__
```

You could get the declaration and return values of API by the header file corresponding to *E4RUInit.dll*. The following is to explain the details on the return values.

Return Value	Micro Definition	Comment
0x00000000	E4RU_SUCCESS	Execution is succeeded
0xE0002002	E4RU_INVALID_PARAMETER	Ineffective parameters
0xE0002003	E4RU_INSUFFICIENT_BUFFER	Insufficient buffer memory
0xE0002004	E4RU_DEVICE_NOT_FOUND	Failed to find device
0xE0002101	E4RU_INVALID_DEVICE_VERSION	Invalid device version
0xE0002101	E4RU_INVALID_DEVICE_VERSION	Invalid device version
0xE0002102	E4RU_OPEN_DEVICE_ERROR	Failed to open device
0xE0002103	E4RU_INSUFFICIENT_MEMORY	Insufficient memory
0xE0002104	E4RU_CREATE_DIR_ERROR	Failed to create the directory
0xE0002105	E4RU_ERASE_DIR_ERROR	Failed to empty the directory
0xE0002106	E4RU_VERIFYPIN_ERROR	Failed to verify the Developer PIN
0xE0002107	E4RU_WRITEFILE_ERROR	Failed to write the file into the device
0xE0002108	E4RU_INSUFFICIENT_DIR	Insufficient memory of specified directory
0xE0002109	E4RU_INVALID_DIRNAME	Invalid directory name
0xE000210A	E4RU_UPDATE_FILE_OVERSIZE	Updating file out of boundary
0xE000210B	E4RU_GET_FILE_INFO_ERROR	Failed to obtain the property of files in device
0xE000210C	E4RU_MISMATCH_FILE_TYPE	The file type in dongle does not match
0xE000210D	E4RU_GET_DEVICE_TYPE_ERROR	Failed to obtain the device type
0xE000210E	E4RU_MISMATCH_DEVICE_TYPE	The device type does not match
0xE000210F	E4RU_GET_AVAILABLE_SPACE_ERROR	Failed to obtain the remaining memory size of the current directory

Table 1

API of Initialization Lib

e4ru_CreateIssuerDevice

Create the Issuer Dongle

```
DWORD WINAPI e4ru_CreateIssuerDevice(
        IN      PDONGLE_CONTEXT     pS4Ctx,
        IN      BYTE                *pDevPin,
        IN      BYTE                *pTDESA
);
```

Parameters:

pS4Ctx	[in]DONGLE_CONTEXT struct pointer, aiming at the device.
pDevPin	[in] Pointer to the Developer PIN (24 bytes) of root directory. When the input is NULL, then the pointer aims at the default Developer PIN “123456781234567812345678”.
pTDESA	[in] The pointer to TDES encryption key (16 bytes) in hex. This value cannot be NULL.

Return values:
E4RU_SUCCESS or error code

Remarks:
Creating the Issuer Dongle will empty and rebuild the root directory; therefore the Developer PIN of root directory of created Issuer Dongle is the default value.

If *pS4Ctx* is NULL, the operation of enumerating and opening device can be accomplished internally by the function.

Requirement:
Hardware Version: Local v2.3 or above

e4ru_CreateClientDevice

Create the User Dongle

```
DWORD WINAPI e4ru_CreateClientDevice(
        IN      PDONGLE_CONTEXT     pS4Ctx,
        IN      LPCSTR              lpDirectory,
        IN      DWORD               Size,
        IN      DWORD               dwEraseFlag,
        IN      BYTE                *pDevPin,
        IN      BYTE                *pTDESA,
        IN      BYTE                ModuleNum,
        IN      BYTE                ErrorMax
);
```

Parameters:

pS4Ctx — [in]DONGLE_CONTEXT struct pointer, aiming at the device.

lpDirectory — [in] Pointer to the specified directory. This function only supports the root directory and first-level sub-directory which is in the following format:

- Root Direcory: ”\”.
- First-level Sub-directory: ”\xxxx”. “xxxx”must be in hex.

If the User Dongle to be created is the network type, this parameter will be non-effective.

Size — [in] It specifies the size of the operation directory. Only when the root directory is specified, the value could be NULL, otherwise, it will be illegal, and the recommended value is 5120 bytes.

dwEraseFlag — [in] Flag value, click here for detail. If the User Dongle to be created is the network type, this parameter will be non-effective.

pDevPin — [in] Pointer to the Developer PIN (24 bytes)

- When the input value is NULL, the Developer PIN is default value.
- When the specified directory does not exist in device:

If the root directory is specified, the input of pointer should be NULL, or the content (“123456781234567812345678”) being pointed.

	If the sub-directory is specified, then the content being pointed must be the Developer ID of its upwards directory. And after the successful creation, the Developer PIN of this sub-directory is the default value. • When the specified directory does exist in device: If *dwEraseFlag* is E4RU_ERASE_DIR, then after the successful creation, the Developer PIN of specified directory will be the default value. If *dwEraseFlag* is E4RU_UPDATE_FILE, then after the successful creation, the Developer PIN of specified directory will not change.
pTDESA	[in] Pointer to TDES encryption key (16 bytes) in hex. This value cannot be NULL.
ModuleNum	[in] Max number of updating module, ranging 1~255
ErrorMax	[in] Max number of operating errors, ranging 1~255

Return values:
E4RU_SUCCESS or error code

Remarks:
• If *pS4Ctx* is NULL, the operation of enumerating and opening device can be accomplished internally by function.

• Once the function is successfully executed, the value of *ModuleNum* and *ErrorMax* cannot be changed.

• In this function, if the first-level sub-directory is selected to build, then its Developer PIN is the default value. It is recommended to set the Developer PIN different from the default.

・Flag Value:

Value	Micro Definition	Comment
0x00000000	E4RU_ERASE_DIR	Empty the specified directory
0x00000001	E4RU_UPDATE_FILE	Does not empty the specified directory, only overwrite the original files

Table 2

Requirement:
Hardware Version: Local v2.3 or above, Network v2.0.5 or above

e4ru_ReleaseFile

Release the hardware program and the device information file.

```
DWORD WINAPI e4ru_ReleaseFile(
        IN        BYTE                  *pTDESA,
        IN        BYTE                  ModuleNum,
        IN        BYTE                  ErrorMax,
        IN OUT    DWORD                 *pdwIssuerLen,
        IN OUT    DWORD                 *pdwClientLen,
        IN OUT    DWORD                 *pdwInfoLen,
        OUT       BYTE                  *pIssuerContent,
        OUT       BYTE                  *pClientContent,
        OUT       BYTE                  *pInfoContent
);
```

Parameters:

pTDESA	[in] Pointer to TDES encryption key (16 bytes) in hex cannot be NULL.
ModuleNum	[in] Max amount of modules, ranging 1~255
ErrorMax	[in] Max amount of errors, ranging 1~255
pdwIssuerLen	[in][out] Pointer to the length of hardware program of Issuer Dongle.
pdwClientLen	[in][out] Pointer to the length of hardware program of User Dongle.
pdwInfoLen	[in][out] Pointer to the length of the information file of User Dongle.
pIssuerContent	[out] Pointer to the content of hardware program of Issuer Dongle.
pClientContent	[out] Pointer to content of hardware program of User Dongle.
pInfoContent	[out] Pointer to content of information files of User Dongle.

Return values:

E4RU_SUCCESS or error code, click here for details.

Remarks:

・When one of parameters *pIssuerContent*, *pClientContent*, *pInfoContent* has NULL as input, after the execution, no files are released. The parameter only returns the actual size of the corresponding files.

・When none of parameters *pIssuerContent*, *pClientContent*, *pInfoContent* has NULL as input, after the successful execution, the files are released. The parameter returns the content of the corresponding files.